J1MI2013: Algorithmes et Programmes: feuille 6

Tableaux(3)

Travaux pratiques

Récupérer depuis le site de l'UE l'archive `tp06.tar.gz` et désarchivez-le. Placez vous ensuite dans le répértoire `tp06`.
Les fonctions demandées dans les exercices 1 à 4 ont été spécifiées dans la feuille de TD 6.

Exercice 1. Compléter le fichier `testInserer.c` avec la fonction `inserer`.
Compiler et exécuter avec la ligne de commande :
./testInserer 1 1 1 1 1

Analyser la fonction `main` et les résultats.

Exercice 2.
Dans le fichier `testInsererOrdonne.c` écrire la fonction `insererOrdonne`. Compiler et exécuter avec la ligne de commande :
./testInsererOrdonne 2 4 6 8 1 3 5 7 9

Analyser la fonction `main` et les résultats.

Exercice 3.
Dans le fichier `testSuppressions.c` écrire les fonctions `supprimer` et `supprimerPremiereOccurrence`.
Le fichier fourni initialise un tableau de 10 cases avec des entiers et appelle les fonctions avec différents arguments. Analyser les résultats.

Exercice 4. Dans le fichier `testRotations.c` :
1. écrire et tester la fonction `rotationAdroite`.
2. écrire et tester deux versions de la fonction qui effectue une rotation à droite de `k` positions sur les `n` premiers éléments de `t`.
   Comment éviter d'effectuer des recopies inutiles si $k > n$?

On remarquera que le programme nécessite qu'on fournisse la valeur de `k` en argument de la ligne de commande (le tableau sur lequel effectuer les tests étant fixé).

Exercice 5. Dans le fichier `testSections.c` écrire une fonction `estSection(t, nt, s, ns)` qui prend en paramètre deux tableaux d'entiers `t` et `s` de taille respective `nt` et `ns` et qui renvoie `true` si `s` correspond à une section du tableau `t`, `false` sinon.
Une section est une suite éventuellement vide d'éléments contigus dans un tableau. Par exemple, cette fonction renvoie `true` pour `t = [5, 1, 2, 3, 1, 2, 1]` et `s = [1, 2]`. Elle renvoie `false` pour `t = [1, 2, 3, 4, 1]` et `s = [1, 3]`.

Exercice 6. En modifiant la fonction précédente, ajouter au fichier `testSections.c` une nouvelle fonction `nombreSections(t, nt, s, ns)` qui calcule et retourne le nombre de fois où le tableau non vide `s` apparaît comme section du tableau `t`.
Par exemple, le tableau `[1, 2]` correspond à deux sections du tableau `[5, 1, 2, 3, 1, 2, 1]`.
Autre exemple, le tableau `[1, 2, 1]` correspond à deux sections du tableau `[5, 1, 2, 1, 2, 1]`.
Dans ce dernier cas, les deux sections se chevauchent :
`[5, 1, 2, 1, 2, 1]` et `[5, 1, 2, 1, 2, 1]`.